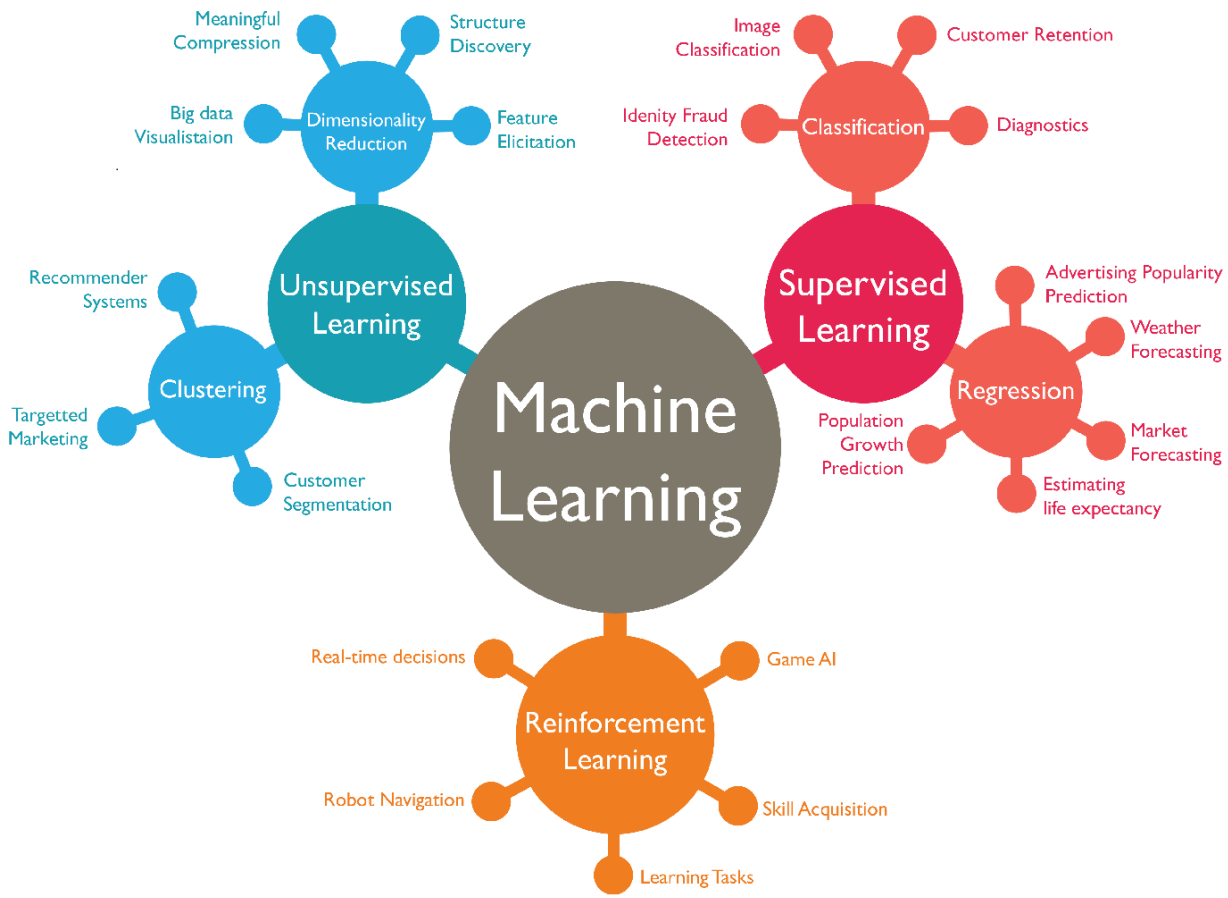
هوش مصنوعی و یادگیری ماشین با تمرکز بر شبکه های عصبی کانولوشنی



آشنایی کلی با ماشین لرنینگ:

هنگام صحبت در مورد الگوریتم‌های Machine Learning (یادگیری ماشین)، ما با دسته‌بندی‌های مختلفی روبرو می‌شویم. الگوریتم‌های یادگیری ماشین می‌توانند به چندین دسته تقسیم شوند. در زیر چند دسته اصلی آورده شده‌اند:

۱- یادگیری نظارت‌شده:(Supervised Learning)

- Regression Algorithms: این الگوریتم‌ها برای پیش‌بینی یک مقدار عددی (مانند قیمت خانه) استفاده می‌شوند. مثال : Linear Regression، Polynomial Regression.

- Classification Algorithms: این الگوریتم‌ها برای تخمین دسته یا برچسب خروجی بر اساس ویژگی‌های ورودی هستند. مثال: Logistic Regression ،. Decision Trees ،. Support Vector Machines.

۲- یادگیری بدون نظارت:(Unsupervised Learning)

- Clustering Algorithms: این الگوریتم‌ها به گروه‌بندی داده‌ها بدون داشتن برچسب می‌پردازند. مثال: K-Means ،. Hierarchical Clustering.

- Dimensionality Reduction Algorithms: این الگوریتم‌ها به کاهش تعداد ویژگی‌ها یا ابعاد داده‌ها می‌پردازند. مثال: Principal Component Analysis (PCA).

۳- یادگیری تقویتی :(Reinforcement Learning)

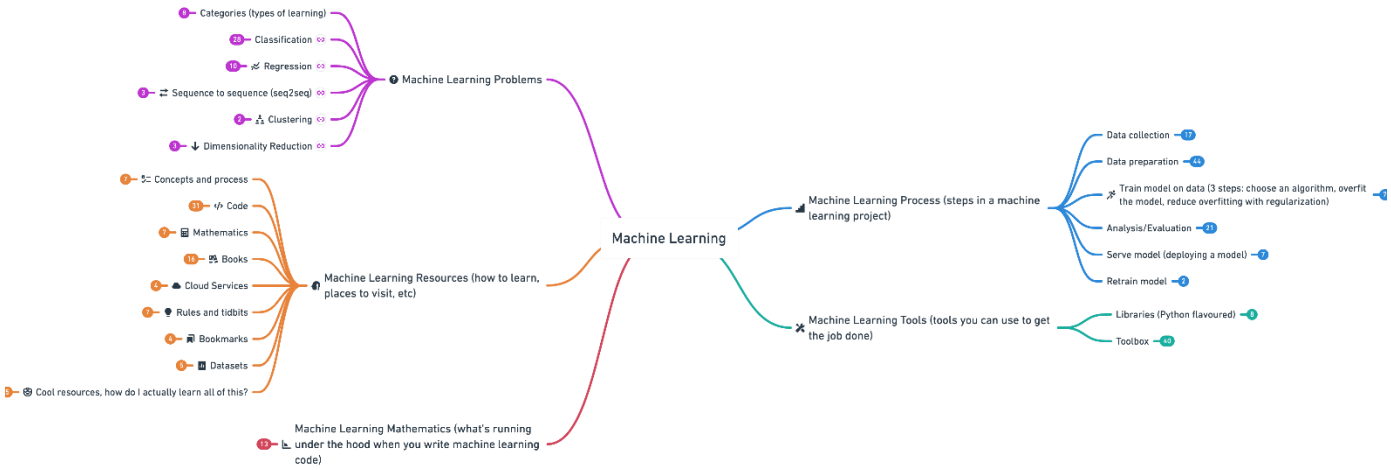
- در این نوع یادگیری، یک عامل (agent) تصمیم‌هایی می‌گیرد تا بهترین عمل را در یک محیط مشخص برای به دست آوردن پاداش (یا جریمه) انجام دهد. مثال: Q-Learning ،. Deep Q Network (DQN).

۴- یادگیری نیمه-نظارت‌شده:(Semi-Supervised Learning)

- این دسته شامل الگوریتم‌هایی است که بخشی از داده‌ها دارای برچسب است و بخشی دیگر بدون برچسب.

۵- یادگیری نظارت شده به صورت مقابله‌ای:(Adversarial Machine Learning)

- مثلاً Generative Adversarial Networks (GANs) که برای تولید داده‌های جدید و واقع‌گرایانه استفاده می‌شوند.



همچنین، در هر یک از این دسته‌ها می‌توان الگوریتم‌های مختلف دید که بر اساس موارد خاصی مانند نوع داده‌ها، مسئله‌های مشخص، یا شرایط ویژه انتخاب می‌شوند. این الگوریتم‌ها ممکن است از روش‌های آموزش مختلف مانند Gradient Descent یا Stochastic Gradient Descentهم استفاده کنند.

Roadmap وارد شدن به عرصه ی یادگیری ماشین:

وارد شدن به عرصه Machine Learning (یادگیری ماشین) می‌تواند یک مسیر جذاب و مفید باشد. در ادامه یک road map برای شروع و پیشرفت در این حوزه را ذکر می‌کنم:

۱. پایه‌های ریاضی و آمار:

- مطالعه ریاضیات پایه شامل جبر خطی، احتمالات و آمار می‌تواند بسیار مفید باشد.

۲. برنامه‌نویسی:

- یادگیری یک زبان برنامه‌نویسی مانند Python یا R. در بسیاری از پروژه‌های Machine Learning از Python استفاده می‌شود.

۳. آشنایی با مفاهیم Machine Learning:

- خواندن کتاب‌ها و منابع آموزشی مقدماتی در زمینه Machine Learning.

- انجام دوره‌های آموزشی آنلاین از منابعی مثل Coursera، edX، Udacity. مثلاً دوره Machine Learning پروفسور Andrew Ng در Coursera.

۴. پروژه‌های عملی:

- شروع به انجام پروژه‌های عملی. این کمک می‌کند تا مفاهیم یادگرفته شده را در عمل تجربه کنید.

۵. آشنایی با کتابخانه‌ها و ابزارها:

- آشنایی با کتابخانه‌های Machine Learning معروف مانند TensorFlow و PyTorch.

- آشنایی با ابزارهای مدیریت داده مانند Pandas و NumPy.

۶. عمیق شدن در Deep Learning:

- مطالعه و آموزش در زمینه شبکه‌های عصبی و Deep Learning.

- انجام پروژه‌های Deep Learning.

۷. آشنایی با مسائل خاص:

- تعیین یک حوزه خاص در Machine Learning که به آن علاقه دارید (مثل پردازش تصویر، پردازش زبان طبیعی و غیره).

۸. شرکت در جامعه Machine Learning:

- مشارکت در انجمن‌ها، کنفرانس‌ها و گروه‌های آنلاین مرتبط با Machine Learning.

۹. آموزش مداوم:

- بازنگری مطالب جدید، پیشرفت‌های علمی و تکنولوژی در عرصه Machine Learning.

۱۰. ایجاد پروفایل شخصی:

- ایجاد یک پروفایل در وب‌سایت‌های مثل GitHub با اشتراک گذاری پروژه‌های شخصی و نمونه کارهای خود.

۱۱. پیشرفت شعف:

- مطمئن شدن از پیشرفت و به‌روز بودن با تحولات حوزه Machine Learning.

۱۲. شروع به کار:

- ارسال رزومه به شرکت‌ها و شروع به جستجوی فرصت‌های شعف‌انگیز در حوزه Machine Learning.

نکته مهم:

هیچ مسیری یکتا برای ورود به عرصه Machine Learning وجود ندارد و این road map تنها یک راهنمای کلی است. شما می‌توانید مسیر خود را با توجه به علایق و اهداف شخصی خود شکل دهید.

تفاوت تحلیلگر داده، دانشمند داده و مهندس داده:

تحلیلگر داده، دانشمند داده و مهندس داده، هر سه نقش مهمی در زمینه داده و هوش مصنوعی دارند، اما هرکدام وظایف و مسئولیت‌های متفاوتی دارند. در زیر به طور مختصر توضیح داده شده است:

تحلیلگر داده:

تحلیلگر داده مسئول ترجمه داده‌ها به اطلاعات و الگوهای مفید است. وظیفه او تحلیل داده‌ها و استخراج اطلاعات مفید برای تصمیم گیری است. او معمولاً دارای تخصص در آمار و تحلیل داده‌ها بوده و از ابزارهای مختلفی مانند Excel, SQL و ابزارهای تحلیل داده آماری استفاده می‌کند. تحلیلگر داده باید قادر به تعبیه الگوها و روابط در داده‌ها و همچنین تسلط بر تکنیک‌های مختلف تجزیه و تحلیل داده‌ها باشد.

دانشمند داده:

دانشمند داده در واقع یک ترکیبی از تحلیلگر داده و مهندس داده است که توانایی‌های تحلیل داده و همچنین توانایی برنامه‌ریزی و مهندسی سیستم‌های داده‌ای را دارد. علاوه بر تحلیل داده‌ها و استخراج الگوها، او باید توانایی پیاده‌سازی و تجزیه و تحلیل سیستم‌های داده را نیز داشته باشد. دانشمند داده ممکن است از ابزارهای برنامه‌نویسی و زبان‌های برنامه‌نویسی برای پیاده‌سازی مدل‌های یادگیری ماشین یا سیستم‌های داده‌ای استفاده کند. (در حال حاضر تمرکز اصلی در این پروژه بر اساس فعالیت های این دسته از افرادی است که به طور مستقیم با هوش مصنوعی و دانش داده سر و کله می زنند.)

مهندس داده:

مهندس داده مسئول ساختاردهی و مدیریت داده‌ها است. او تخصص برنامه‌نویسی و مهندسی نرم‌افزار را دارد و به طراحی و اجرای سیستم‌های داده‌ای مانند پایگاه داده‌ها، سیستم‌های توزیع شده و فرآیندهای استخراج و تحلیل داده می‌پردازد. مهندس داده علاوه بر ایجاد ساختارها و بسترهای داده‌ای، مسئول امنیت و مدیریت کیفیت داده در سیستم‌های داده‌ای نیز می‌باشد.

در کل، تحلیلگر داده بیشتر به تحلیل و استخراج اطلاعات تمرکز دارد، دانشمند داده دارای مهارت‌های تحلیل و برنامه‌ریزی و مهندس داده بیشتر به زمینه بنیادین فناوری داده و مهندسی آن تخصص دارد.

به چه کسی دانشمند داده می‌گویند؟ دانشمند داده باید چه دانشی داشته باشد؟

یک دانشمند داده حرفه‌ای باید توانایی‌ها و دانش فنی گسترده‌ای داشته باشد. در زیر تعدادی از مهم‌ترین مواردی که یک دانشمند داده باید بداند را بررسی می‌کنیم:

۱. برنامه‌نویسی: تسلط بر زبان‌های برنامه‌نویسی مانند پایتون، R، جاوا یا سی‌پلاس‌پلاس و توانایی ایجاد و توسعه کدهای تحلیل داده.

۲. پایگاه داده: شناخت دقیقی از سیستم‌های مدیریت پایگاه داده و زبان SQL برای پرس و جوی داده‌ها.

۳. **یادگیری ماشین و یادگیری عمیق**: تسلط بر الگوریتم‌های یادگیری ماشین، شبکه‌های عصبی و عمیق، ارزیابی مدل‌ها و پیاده‌سازی آنها بر روی داده‌ها.

۴. پردازش و تحلیل داده: توانایی پیش‌پردازش داده، تحلیل و تفسیر نتایج به دست آمده از مدل‌های یادگیری ماشین، و توانایی بهینه‌سازی عملکرد مدل‌ها.

۵. داده‌کاوی: توانایی کشف الگوهای مخفی و روابط پنهان در داده‌ها و ارائه روش‌های بهبود کارایی و پیش‌بینی دقیق.

۶. اصول و روش‌های آماری: مفاهیم پایه‌ای آماری و تجزیه و تحلیل داده‌های آماری برای استخراج الگوها و اطلاعات مفید از داده.

۷. تجربه صنعتی: توانایی ارائه راه‌حل‌های داده‌محور برای مسائل صنعتی، توانایی همکاری با تیم‌های فنی و توانایی ارتباط با مدیران و تصمیم‌گیران.

۸. دانش تخصصی: تسلط بر حوزه‌های خاص مانند تحلیل مالی، پردازش زبان طبیعی، بینایی ماشین و ...

۹. کار با ابزارهای داده‌ای: توانایی استفاده از ابزارهای متنوع و پرکاربرد مانند TensorFlow, scikit-learn, Hadoop, Spark و ...

۱۰. اخلاقیات داده: آگاهی از مسائل اخلاقی و حریم خصوصی مرتبط با استفاده از داده‌ها و مدیریت آنها.

همچنین، یک دانشمند داده حرفه‌ای باید دارای مهارت‌های مدیریتی، توانایی حل مسائل، ارتباطات بین فردی بالا و توانایی کار در تیم‌های چند رشته‌ای و همچنین داشتن مدرک تحصیلی مرتبط به عنوان پژوهشگر یا دانشجو می‌باشد.

یادگیری ماشین چیست؟

ماشین لرنینگ (Machine Learning) یک زیرشاخه از هوش مصنوعی (Artificial Intelligence) است که به کامپیوترها امکان یادگیری بدون نیاز به برنامه‌ریزی صریح می‌دهد. در واقع، با ماشین لرنینگ، کامپیوترها قابلیت یادگیری از داده‌ها و تجربیات خود را دارند و تصمیم‌گیری‌ها و پیش‌بینی‌های خود را بر اساس این یادگیری انجام می‌دهند.

یادگیری ماشین یک زمینه از هوش مصنوعی است که به الگوریتم ها و مدل هایی که به وسیله داده ها آموزش داده می‌شوند، اشاره دارد. اصطلاح "یادگیری ماشین" به آنجا اشاره دارد که چگونه یک سیستم ماشینی توانایی بهبود عملکرد خود را از طریق تجربه و تعامل با داده ها نشان دهد، به جای این که به صورت صریح برنامه ریزی شود. یادگیری ماشین در اصل به دسته ای از الگوریتم ها و مدل ها اطلاق می‌شود که از داده های ورودی یاد می‌گیرند و با استفاده از آن‌ها می‌توانند پیش‌بینی کننده‌ها، الگوها و روابط پنهان در داده ها را مدل کنند. این الگوریتم ها و مدل ها می‌توانند در بسیاری از حوزه های کاربردی مانند تشخیص تصاویر، پرکاربرد است.

برای تسلط بر یادگیری ماشین باید چه کار کنم؟

برای یادگیری ماشین لرنینگ، می‌توانید از روش‌های زیر استفاده کنید:

۱. مطالعه مقدماتی: شروع کنید با مطالعه مفاهیم پایه ماشین لرنینگ و یادگیری ماشین. مفاهیمی مانند انواع مدل‌ها، توابع هزینه و روش‌های بهینه‌سازی را بیاموزید.

۲. دوره‌های آموزشی و منابع آموزشی آنلاین: امروزه، بسیاری از وب‌سایت‌ها و پلتفرم‌های آموزشی دوره‌های عالی در زمینه ماشین لرنینگ ارائه می‌دهند. مثال‌هایی از این منابع شامل Coursera، Udemy، Khan Academy و ... می‌باشد.

۳. کتابخانه‌های ماشین لرنینگ: یادگیری ماشین و یادگیری عمیق می‌تواند با استفاده از کتابخانه‌هایی مانند TensorFlow, scikit-learn, Keras و PyTorch آموزش داده شود. مطالعه مستندات و آموزش‌های موجود درباره این کتابخانه‌ها می‌تواند به شما کمک کند تا نحوه استفاده از آنها را بیاموزید.

۴. پروژه‌های عملی: یکی از بهترین راه‌ها برای یادگیری ماشین لرنینگ، شرکت در پروژه‌های عملی و پیاده‌سازی مدل‌های واقعی است. با تجربه عملی و مواجهه با چالش‌های واقعی، شما می‌توانید مهارت‌های خود را در این زمینه تقویت کنید.

۵. گروه‌های مطالعه و انجمن‌های تخصصی: پیوستن به گروه‌های مطالعه و انجمن‌های تخصصی در موضوعات ماشین لرنینگ، برنامه‌نویسی و علوم داده، می‌تواند به شما کمک کند تا از تجربیات دیگران بهره ببرید و از جدیدترین تکنیک‌ها و شیوه‌های کاربردی مطلع شوید.

مهمتر از همه، توجه به تمرین و انجام تجربه‌های عملی در حوزه ماشین لرنینگ و یادگیری عمیق است. این باعث تقویت مهارت‌های عملی شما و بهبود استفاده از ابزارها و روش‌های مختلف آموزشی می‌شود.

مراحل عملکرد ماشین لرنینگ:

۱. تجمیع داده:

در ابتدا، داده‌هایی که به کامپیوتر ارائه می‌شوند، جمع‌آوری و ذخیره می‌شوند. این داده‌ها ممکن است از اندازه‌های مختلف و انواع مختلفی باشند.

۲. انتخاب مدل:

مدل‌ها نوعی الگوریتم هستند که براساس داده‌ها یاد می‌گیرند. انتخاب مدل مناسب برای مسئله مورد نظر یک گام اساسی است.

۳. آموزش مدل:

در مرحله آموزش، مدل با استفاده از داده‌های ورودی و خروجی متناظر، پارامترهای خود را به‌روزرسانی می‌کند تا بتواند بهترین تطابق را با ورودی‌ها ایجاد کند.

۴. ارزیابی مدل:

-مدل ارزیابی می‌شود تا مشخص شود چقدر در پیش‌بینی داده‌های جدید عملکرد خوبی دارد. این مرحله بررسی می‌کند که مدل به درستی یاد گرفته است یا خیر.

۵. پیش‌بینی و استفاده از مدل:

پس از آموزش مدل، آن را می‌توان برای پیش‌بینی یا تصمیم‌گیری در مورد داده‌های جدید استفاده کرد. مدل می‌تواند به طور خودکار الگوها را در داده‌های جدید تشخیص دهد.

کاربردهای ماشین لرنینگ:

- **پردازش تصویر**: تشخیص اشیاء، توصیف تصاویر و تشخیص چهره.
- **پردازش زبان طبیعی**:ترجمه ماشینی، تولید متن خودکار، تحلیل احساسات متن.
- **پیش‌بینی و تحلیل:** پیش‌بینی قیمت‌ها، تحلیل بازارهای مالی، پیش‌بینی سری‌های زمانی.
- **پزشکی:** تشخیص بیماری‌ها، پیش‌بینی نتایج درمان، شناسایی نقاط ضعف در تصاویر پزشکی.
- **مهندسی مخابرات:** بهبود عملکرد شبکه‌های ارتباطی و مدیریت ترافیک.
- **روباتیک:** یادگیری مهارت‌های حرکتی و تصمیم‌گیری‌های یک ربات.
- **تشخیص الگو و درسته بندی:** ماشین لرنینگ برای تشخیص الگوها و دسته‌بندی داده‌ها در حوزه‌های مختلف استفاده می‌شود، از جمله تشخیص پدیده های جغرافیایی، تشخیص الگوهای پزشکی، یا حتی تشخیص الگوهای مالی.
- **بازاریابی و تبلیغات:** برای پیشنهاد به مشتریان، تحلیل رفتار مشتریان، پیش بینی بازار و بهبود کمپین‌های تبلیغاتی

مدل ها و الگوریتم های معروف یادگیری ماشین

مدل‌های یادگیری ماشین بسیار متنوع هستند و برای مسائل مختلف از آنها استفاده می‌شود. در زیر چند مدل معروف یادگیری ماشین را معرفی کرده و توضیح داده‌ام:

۱. Linear Regression (رگرسیون خطی):

یکی از ساده‌ترین مدل‌ها در یادگیری ماشین است. این مدل برای پیش‌بینی یک متغیر وابسته مانند قیمت یک محصول بر اساس متغیرهای مستقل استفاده می‌شود. هدف از این مدل، یافتن یک خطی که بیشترین تطابق را با داده‌ها داشته باشد.

۲. Logistic Regression (رگرسیون لجستیک):

این مدل برای مسائل دسته‌بندی دو دسته‌ای استفاده می‌شود. به عبارت دیگر، این مدل به ما کمک می‌کند تا احتمال تعلق یک نمونه به یک دسته را پیش‌بینی کنیم.

۳. Decision Trees (درخت تصمیم):

این مدل از ساختار درختی برای تصمیم‌گیری استفاده می‌کند و برای دسته‌بندی و پیش‌بینی استفاده می‌شود. این مدل به صورت یک درخت تصمیم با گره‌های تصمیم و شاخه‌های گره‌ها مدل می‌شود.

۴. Random Forest:

Random Forest یک مدل انبوهی است که از ترکیب چندین درخت تصمیم برای ایجاد یک مدل قوی‌تر استفاده می‌کند. این مدل برای دسته‌بندی و پیش‌بینی استفاده می‌شود و مقاومیت خوبی در برابر برازش بالا و اورفیت دارد.

۵. Support Vector Machines (ماشین‌های بردار پشتیبان):

SVM یک مدل دسته‌بندی است که با ماکسیمم کردن حاشیه بین دو دسته از داده‌ها کار می‌کند و برای مسائل دسته‌بندی خطی و غیرخطی استفاده می‌شود.

۶. Neural Networks (شبکه‌های عصبی):

این مدل مستند به ساختار عصبی انسان است و برای حل مسائل پیچیده از جمله تشخیص تصویر، پردازش زبان طبیعی، اعمال توابع پیچیده و ... استفاده می‌شود. شبکه‌های عصبی می‌توانند شامل چندین لایه مختلف با نورون‌های مختلف باشند.

این تنها چند مدل معروف یادگیری ماشین بودند و هر یک از این مدل‌ها به عنوان یک ابزار برای حل مسائل مختلف قابل استفاده هستند.

یادگیری عمیق چیست؟

یادگیری عمیق یک زیرشاخه از یادگیری ماشین است که شامل استفاده از شبکه‌های عصبی ژرف برای یادگیری و تفسیر الگوها از داده‌ها است. در واقع، یادگیری عمیق به ماشین‌ها توانایی می‌دهد تا الگوها و ویژگی‌های پیچیده‌تری را از داده‌ها استخراج کنند.

شبکه‌های عصبی ژرف، یک نوع ساختار محاسباتی هستند که به تقلید از ساختار مغز انسان استوارند. این شبکه‌ها از لایه‌های متعددی از نورون‌ها تشکیل شده‌اند که اطلاعات را از ورودی‌ها به صورت موازی پردازش می‌کنند و ویژگی‌های پیچیده‌تر را استخراج می‌کنند.

یادگیری عمیق در بسیاری از حوزه‌های سنتی یادگیری ماشین تاثیرگذار بوده است، از جمله تشخیص تصاویر، تشخیص گفتار، ترجمه ماشینی، پردازش زبان طبیعی، بازیابی اطلاعات، و بسیاری دیگر. واحدهای عملکردی یادگیری عمیق در تشخیص الگوها، تفسیر داده‌ها، خودکارسازی و پیش‌بینی دقیق است.

مراحل عملکرد یادگیری عمیق:

مراحل عملکرد یادگیری عمیق شامل مراحل زیر می‌شود:

۱. تعریف مسئله:

در این مرحله باید مسئله مورد نظر که می‌خواهیم با استفاده از یادگیری عمیق حل کنیم، به وضوح تعریف شود. این شامل تعیین وظیفه‌ای که می‌خواهیم مدل حل کند و نوع داده‌هایی که برای آموزش مدل مورد استفاده قرار می‌دهیم، می‌شود.

۲. جمع‌آوری داده:

در این مرحله، داده‌های مورد نیاز برای آموزش مدل جمع‌آوری می‌شود. این می‌تواند شامل تصاویر، متن، صدا یا هر نوع داده‌ی دیگری باشد که برای حل مسئله ما نیاز است.

۳. پیش‌پردازش داده:

قبل از اعمال داده‌ها به مدل یادگیری عمیق، باید داده‌ها را پیش‌پردازش کنیم. این شامل تمییز دادن، تقسیم بندی، مقیاس دادن و تمیزکاری از داده‌ها می‌شود تا مدل بهتری را آموزش دهیم.

۴. طراحی مدل:

در این مرحله، معماری مدل یادگیری عمیق انتخاب می‌شود. این شامل تعیین تعداد لایه‌ها و نوع لایه‌ها مانند لایه‌های پیچشی، لایه‌های راهبردی، و لایه‌های متصل کامل است.

۵. آموزش مدل:

در این مرحله، مدل با استفاده از داده‌های آموزشی آموزش داده می‌شود تا بتواند الگوها و ویژگی‌های مهم را از داده‌ها استخراج کرده و به یاد بگیرد.

۶. ارزیابی مدل:

پس از آموزش مدل، مدل باید با استفاده از داده‌های آزمایشی ارزیابی شود تا ببینیم چقدر مدل قادر به پیش‌بینی صحیح و دقیق است.

۷. استفاده از مدل:

در مرحله نهایی، مدل آموزش داده شده برای استفاده در مسئله واقعی استفاده می‌شود. این شامل استفاده از مدل برای پیش‌بینی‌ها، تصمیم گیری‌ها و حل مسائل عملی می‌شود.

همچنین ارتقا و بهبود مدل‌ها از طریق فرایندهایی مانند تیونینگ هایپرپارامتر، واکشی یادگیری (Transfer Learning)و... نیز جزو مراحل اصلی یادگیری عمیق می‌باشد.

برای تسلط بر یادگیری عمیق باید چه کار کنم؟

برای یادگیری دیپ لرنینگ (یادگیری ژرف)، می‌توانید از راهنمایی‌های زیر استفاده کنید:

۱. مفاهیم پایه را بیاموزید:

درک مفاهیم پایه یادگیری عمیق از جمله شبکه‌های عصبی، لایه‌ها، توابع فعال‌سازی، توابع هزینه و بهینه‌سازها، اولین گام مهمی برای ورود به دنیای دیپ لرنینگ است.

۲. مطالعه دوره‌های آموزشی:

یکی از راه‌های موثر برای یادگیری دیپ لرنینگ، مطالعه دوره‌های آموزشی مانند دوره‌های Coursera و Udacity در زمینه یادگیری عمیق است که توسط متخصصان معتبر تدریس می‌شود.

۳. مطالعه کتب و منابع الکترونیکی:

کتب و منابع الکترونیکی متعددی در زمینه یادگیری عمیق وجود دارند که می‌توانند به شما کمک کنند تا از اصول و الگوریتم‌های اصلی یادگیری عمیق مطلع شوید.

۴. پیاده‌سازی پروژه‌های عملی:

یکی از بهترین راه‌ها برای آموزش و یادگیری دیپ لرنینگ، پیاده‌سازی پروژه‌های عملی و واقعی است. این کار به شما کمک می‌کند تا اصول یادگیری عمیق را در عمل تجربه کنید و مهارت‌های عملی خود را تقویت کنید.

۵. مشارکت در انجمن‌ها و گروه‌های مطالعه:

فعالیت در انجمن‌ها و گروه‌های مطالعه در زمینه یادگیری عمیق و ماشین لرنینگ می‌تواند به شما کمک کند تا نحوه استفاده از ابزارها و روش‌های جدید را یاد بگیرید و در ارتباط با افراد دیگری که در این زمینه فعال هستند، قرار بگیرید.

۶. کدنویسی و استفاده از کتابخانه‌های مربوطه:

انجام پروژه‌های عملی با استفاده از کتابخانه‌های مربوط به یادگیری عمیق مانند TensorFlow, Keras, PyTorch و غیره، یک روش عالی برای یادگیری مفاهیم عمیق‌تر یادگیری عمیق است.

۷. مطالعه و شرکت در مسابقات و همایش‌ها:

شرکت در مسابقات و همایش‌های مرتبط با یادگیری عمیق و ماشین لرنینگ می‌تواند شما را به روز نگه دارد و به شما اجازه بدهد تا از دیگران یاد بگیرید.

اهمیت تمرین و تجربه عملی در این حوزه بسیار بالاست. به این دلیل، ایجاد و پیشروی در پروژه‌های مختلف و انجام آزمایش‌های متعدد می‌تواند به شما کمک کند تا مهارت‌های عملی و نگاه کاربردی‌تری را در این حوزه پیدا کنید.

کاربرد های شبکه های عصبی عمیق:

دیپ لرنینگ (یادگیری ژرف) بسیار گسترده‌ای دارد و در بسیاری از حوزه‌های مختلف کاربرد دارد. برخی از کاربردهای اصلی دیپ لرنینگ عبارتند از:

۱. تشخیص تصاویر:

از جمله کاربردهای مهم دیپ لرنینگ، تشخیص و دسته‌بندی تصاویر است. این شامل تشخیص اشیاء در تصاویر، تشخیص چهره، تشخیص اعضاء بدن و بسیاری دیگر می‌شود.

۲. پردازش زبان طبیعی:

دیپ لرنینگ نقش مهمی در پردازش زبان طبیعی دارد، از جمله ترجمه ماشینی، تشخیص و تحلیل احساسات، تولید متن و تشخیص گفتار.

۳. ترجمه ماشینی:

دیپ لرنینگ می‌تواند در ترجمه ماشینی بسیار مؤثر باشد و به طور خاص در ترجمه بین زبان‌های مختلف تاثیرگذار است.

۴. تشخیص الگوها و پترن‌ها:

دیپ لرنینگ می‌تواند در تشخیص الگوها و پترن‌های پیچیده در داده‌ها مؤثر باشد، از جمله تشخیص الگوهای پزشکی، تشخیص الگوهای مالی، و ...

۵. تشخیص موسیقی و صدا:

دیپ لرنینگ به تشخیص و تحلیل موسیقی و صدا نیز مورد استفاده قرار می‌گیرد، از جمله تشخیص خواننده‌ها، تحلیل الگوهای موسیقی، تشخیص گفتار و ...

۶. بازیابی اطلاعات:

در حوزه بازیابی اطلاعات و جستجوی متن، دیپ لرنینگ به تشخیص الگوهای متنی، دسته‌بندی متن، و جستجوی موثر اطلاعات کمک می‌کند.

۷. خودروهای خودران:

دیپ لرنینگ نقش بسیار مؤثری در توسعه فناوری‌های خودروهای خودران و خودرانی ایفا می‌کند و برای تشخیص موانع، تصمیم‌گیری‌های پیشرانه، و انجام وظایف خودروی خودران بسیار مهم است.

۸. پزشکی و داروسازی:

در حوزه پزشکی، دیپ لرنینگ برای تشخیص بیماری‌ها از تصاویر پزشکی، پیش‌بینی تراکم‌های دارویی و پروتئین‌ها، و حتی ایجاد داروهای جدید مورد استفاده قرار می‌گیرد.

۹. صنایع مختلف:

در بسیاری از صنایع مختلف، از جمله صنایع نفت و گاز، کشاورزی، تجارت، تولید، و ...

از طرفی، دیپ لرنینگ به عنوان یک ابزار قدرتمند و چابک برای تشخیص الگوها، پیش‌بینی‌ها و تصمیم‌گیری‌های پیچیده در حوزه‌های مختلف، مورد توجه قرار گرفته و به عنوان یکی از مهمترین فناوری‌های دهه حاضر مورد استفاده قرار می‌گیرد.

مدل ها و الگوریتم های معروف یادگیری عمیق

الگوریتم‌ها و مدل‌های یادگیری عمیق در حوزه هوش مصنوعی و یادگیری ماشین بسیار متنوع هستند. در ادامه چند مدل معروف یادگیری عمیق را معرفی و توضیح می‌دهم:

۱. شبکه‌های عصبی پیچشی: Convolutional Neural Networks – CNNs

CNNها از جمله اصلی‌ترین الگوریتم‌ها برای پردازش تصاویر و تشخیص الگوها در تصاویر هستند. آن‌ها از الگوهای ساختاری مشخصی برای تصاویر استفاده می‌کنند که بهترین عملکرد را در تشخیص اشیاء مانند چهره‌ها، اشیاء و صحنه‌ها ارائه می‌دهند.

۲. شبکه‌های عصبی بازگشتی Recurrent Neural Networks - RNNs

RNNها عمدتاً برای مدل‌سازی داده‌های دنباله‌ای مانند زبان، گفتار و سایر داده‌های زمانی استفاده می‌شوند. آن‌ها از حافظه داخلی برای به خاطر سپردن اطلاعات گذشته در دنباله‌ها استفاده می‌کنند.

۳. شبکه‌های عصبی مکرر

LSTM، یک نوع ویژه از شبکه‌های عصبی بازگشتی است که از مشکل محو شدن گرادیان در RNNها جلوگیری می‌کند و برای مدل‌سازی داده‌های دنباله‌ای به کار می‌رود. LSTM به خاطر سپردن اطلاعات مهم در طول دنباله‌ها شناخته می‌شود.

۴. تبدیل‌دهنده‌های توجه : Transformers

این الگوریتم‌ها اخیراً برای پردازش زبان طبیعی و ترجمه ماشینی بسیار مؤثر بوده‌اند. آن‌ها به دلیل قابلیت پردازش همزمان داده‌های ورودی بسیار بلند به خوبی شناخته شده اند.

۵. شبکه‌های تولید مقدمه‌گرا Generative Adversarial Networks - GANs

GANها برای تولید داده‌های جدید و قابل قبول براساس تصاویر و داده‌های زمانی مورد استفاده قرار می‌گیرند. این شبکه‌ها از رقابت بین دو شبکه (یک مولد و یک تمیزدهنده) برای تولید داده‌های واقعی بهره می‌برند.

۶. شبکه‌های عصبی مکرر توجه Attention-based Recurrent Neural Networks

این الگوریتم‌ها از مکانیسم توجه برای تمرکز بر اجزاء مهم دنباله‌ها استفاده می‌کنند و در حوزه‌هایی از جمله ترجمه ماشینی و پردازش زبان طبیعی مؤثر هستند.

این تنها چند مثال از مدل‌های یادگیری عمیق و شبکه‌های عصبی بودند که در حال حاضر در حوزه‌های مختلف از جمله پردازش تصویر، پردازش زبان طبیعی، تولید محتوا و ... مورد استفاده قرار می‌گیرند.

بینایی ماشین و پردازش تصویر

بینایی ماشین به معنای استفاده از الگوریتم‌ها و مدل‌های یادگیری ماشین برای تشخیص و تفسیر تصاویر و ویدیوها است. این فناوری بر اساس یادگیری عمیق و دیپ لرنینگ برای تشخیص الگوها و ویژگی‌های مختلف در تصاویر استفاده می‌شود. بینایی ماشین به ماشین‌ها امکان تشخیص الگوها، اشیاء، صحنه‌ها و ویژگی‌های مختلف در تصاویر و ویدیوها را می‌دهد.

به عبارت دیگر، بینایی ماشین می‌تواند به ماشین‌ها کمک کند تا بفهمند چه چیزی در تصویر یا ویدیو قرار دارد و بتوانند الگوها و ویژگی‌ها را تشخیص دهند. این تکنولوژی در بسیاری از زمینه‌ها مانند تشخیص چهره، تشخیص اشیاء، خودروهای خودران، پزشکی، امنیت، تشخیص تصادفات، اطلاعات جغرافیایی، کشاورزی و ... استفاده می‌شود.

در واقع، بینایی ماشین به ماشین‌ها امکان می‌دهد تا از تصاویر و ویدیوها به عنوان ورودی داده‌های خام استفاده کنند و الگوها و ویژگی‌های مختلف را تشخیص دهند تا به طور خودکار اطلاعات مورد نیاز را استخراج کرده و تحلیل کنند.

پردازش تصویر به معنای استفاده از الگوریتم‌ها و روش‌های مختلف برای تحلیل و تغییر تصاویر و ویدیوها است. این فرایند شامل استخراج ویژگی‌های مختلف از تصویر، تشخیص الگوها و اشیاء، بهبود کیفیت تصویر، تشخیص و دسته‌بندی اجسام، تشخیص ماهیت اجسام و ... می‌شود.

پردازش تصویر به صورت گسترده در بسیاری از حوزه‌ها مورد استفاده قرار می‌گیرد؛ از پزشکی و پردازش تصاویر پزشکی گرفته تا بینایی ماشین، صنعت، امنیت، روباتیک، واقعیت مجازی، بازی‌های ویدیویی و ... .

با پیشرفت فناوری یادگیری عمیق و دیپ لرنینگ، پردازش تصویر دقت و کارآیی بسیار بالایی کسب کرده و به صورت گسترده در اپلیکیشن‌ها و سیستم‌های هوش مصنوعی مورد استفاده قرار گرفته است. از طرفی، پردازش تصویر می‌تواند به ماشین‌ها کمک کند

تا از تصاویر به عنوان داده‌های ورودی استفاده کنند و اقدامات مختلفی نظیر تشخیص الگوها، تشخیص اشیاء، تشخیص چهره، تشخیص مکان و اندازه اشیاء و ... را انجام دهند.

در کل، پردازش تصویر به تجزیه و تحلیل تصاویر با هدف استخراج اطلاعات مفید، تشخیص الگوها و اشیاء و اعمال تغییرات مختلف بر روی تصاویر می‌پردازد.

CNN:

شبکه‌های عصبی کانولوشنی یا CNN (Convolutional Neural Networks) یک نوع از شبکه‌های عصبی عمیق هستند که به طور خاص برای پردازش تصاویر و ساختارهای داده‌ای دوبعدی مورد استفاده قرار می‌گیرند. این شبکه‌ها به اندازه زیاد در تشخیص الگوها، تصویربرداری، تشخیص اشیاء، وظایف پردازش تصویر و چهره‌گری استفاده می‌شوند. شبکه‌های عصبی پیچشی یا CNN به عنوان یکی از انواع اصلی شبکه‌های عصبی مصنوعی شناخته می‌شوند. این نوع از شبکه‌های عصبی برای پردازش تصاویر و تشخیص الگوها در تصاویر بسیار موثر هستند. CNN ها از ساختارهای مختلفی که مشابه به ساختار قشر بصری انسان است، الهام گرفته‌اند.

معماری اصلی یک CNN شامل لایه‌های پیچشی ((Convolutional Layers، لایه‌های ادغام ((Pooling Layersو لایه‌های کاملا متصل ((Fully Connected Layersمی‌باشد. لایه‌های پیچشی برای استخراج ویژگی‌ها از تصویر استفاده می‌شوند، لایه‌های ادغام برای کاهش ابعاد حجم داده و لایه‌های کاملا متصل از ویژگی‌های استخراج شده برای دسته‌بندی نهایی استفاده می‌کنند.

CNN ها دارای ویژگی‌هایی همچون انتقال پارامترها ((Parameter Sharingدر لایه‌های پیچشی به منظور کاهش تعداد پارامترها و دسته‌بندی دقیق تصاویر هستند. این ویژگی‌ها باعث می‌شوند که CNN ها برای مسائل مربوط به تصاویر مانند تشخیص الگوها، تشخیص اشیاء، تشخیص صحنه و ... بسیار مؤثر باشند.

با پیشرفت فناوری یادگیری عمیق و دیپ لرنینگ، CNNها به طور گسترده در بینایی ماشین، تشخیص چهره، تشخیص شیء و صحنه، خودروهای خودران، پزشکی تصویری و ... استفاده می‌شوند. از آنجا که CNN ها قابلیت خودآموزی دارند، آنها می‌توانند با داشتن مجموعه داده‌های مناسب به طور خودکار الگوها و ویژگی‌های مختلف را در تصاویر تشخیص دهند.

ویژگی‌های CNN:

۱. لایه‌های کانولوشن:

- این لایه‌ها از عملیات کانولوشن برای استخراج ویژگی‌های تصویر استفاده می‌کنند. فیلترها (کرنل‌ها) روی تصویر حرکت کرده و ویژگی‌های مختلف را استخراج می‌کنند.

۲. لایه‌های حشو (Padding):

- برای حفظ اطلاعات در کناره‌های تصویر در طول عملیات کانولوشن، لایه‌های حشو معمولاً به تصویر ابعاد اضافه می‌کنند.

۳. لایه‌های ادغام (Pooling):

- این لایه‌ها برای کاهش ابعاد تصویر و افزایش این توانایی استفاده می‌شوند. ادغام معمولاً با استفاده از ماکسیمم یا میانگین انجام می‌شود.

۴. لایه‌های کاملاً متصل:

- لایه‌هایی که اطلاعات ویژگی‌های استخراج‌شده را به صورت یک بردار ویژگی به لایه‌های خروجی منتقل می‌کنند.

کاربردها:

۱. تشخیص اشیاء:

- CNN به طور گسترده در تشخیص اشیاء در تصاویر مورد استفاده قرار می‌گیرند. این امکان را فراهم می‌کنند تا شبکه‌ها بتوانند اشیاء مختلف را با دقت بالا تشخیص دهند.

۲. تشخیص چهره:

- CNN برای تشخیص چهره‌ها در تصاویر و ویدئوها مورد استفاده قرار می‌گیرد. این تکنولوژی به طور گسترده در سیستم‌های تشخیص چهره و حفاظت از حریم شخصی مورد استفاده قرار می‌گیرد.

۳. ترجمه ماشینی:

- CNN در ترجمه ماشینی نیز به کار می‌روند. مثلاً مدل‌های ترجمه‌ای با استفاده از CNN می‌توانند متن یک زبان را به زبان دیگر ترجمه کنند.

یک مثال از CNN:

هرچند یک پروژه ی کامل راجب CNN، پیوست شده، با این حال برای توضیح بیشتر، یک مثال کوتاهتر اینا ذکر شده تا شما بیشتر با CNN آشنا شوید.

مطمئنا! برای نوشتن یک کد CNN، از کتابخانه‌های محبوب مانند TensorFlow و Keras در پایتون معمولاً استفاده می‌شود. در زیر، یک مثال ساده از یک مدل CNN برای تشخیص اشیاء در تصاویر با استفاده از TensorFlow و Keras آورده شده است.

```python
# Importing necessary libraries
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

# Creating a sequential model
model = Sequential()

# Adding a convolutional layer with 32 filters, kernel size (3,3), and input shape (64,64,3)
model.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))

# Adding a max pooling layer with pool size (2,2)
model.add(MaxPooling2D(pool_size=(2, 2)))

# Adding another convolutional layer with 64 filters and kernel size (3,3)
model.add(Conv2D(64, (3, 3), activation='relu'))

# Adding another max pooling layer
model.add(MaxPooling2D(pool_size=(2, 2)))

# Flattening the matrix to a vector
model.add(Flatten())

# Adding a fully connected layer with 128 neurons
model.add(Dense(128, activation='relu'))

# Adding an output layer with 1 neuron and sigmoid activation (binary classification)
model.add(Dense(1, activation='sigmoid'))

# Compiling the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Displaying the model summary
model.summary()
```

در این مثال:

- از یک `model` Sequential شروع می‌کنیم.
- سه لایه کانولوشن با فیلترهای ۳۲ و ۶۴ تعریف می‌کنیم.
- از لایه‌های ادغام برای کاهش ابعاد تصویر استفاده می‌کنیم.
- با استفاده از لایه Flatten، ماتریس تصویر به بردار تبدیل می‌شود.
- یک لایه کاملاً متصل با ۱۲۸ نورون افزوده می‌شود.
- یک لایه خروجی با ۱ نورون و فعال‌ساز sigmoid اضافه می‌شود.
- مدل با استفاده از تابع بهینه‌سازی "adam" و تابع هزینه "binary_crossentropy" کامپایل می‌شود.

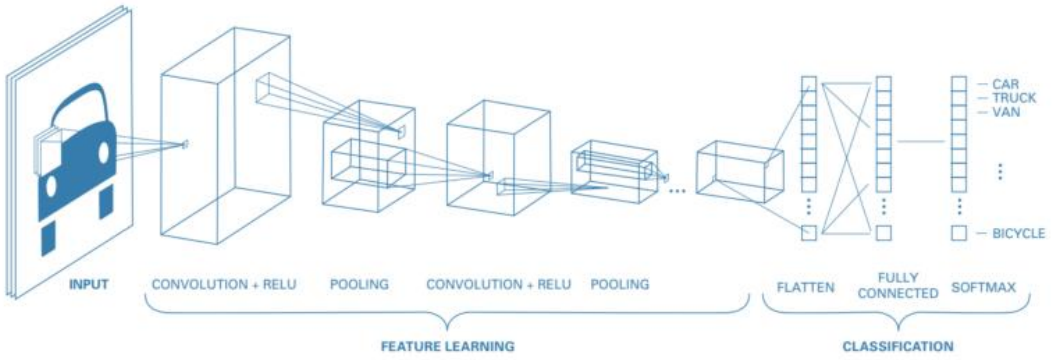
شرح پروژه انجام شده:

در ذیل به شرح پروژه انجام شده می‌پردازیم.

شبکه های عصبی کانولوشنی یا پیچشی(Networks Neural Convolutional) نوعی مدل یادگیری عمیق هستند که به طور گسترده و به خصوص برای پردازش تصویر و ویدیو استفاده میشوند. این مدلها برای تجزیه و تحلیل و شناسایی موثر الگوهای بصری با استفاده از مفهوم Convolution طراحی شده اند .CNN ها معمولا از چندین لایه شامل لایه های پیچشی ، لایه های ادغام و لایه های کاملا متصل تشکیل میشوند .لایه های کانولوشن، فیلترهایی را روی تصویر ورودی اعمال میکنند و ویژگیها را در مکانهای مختلف تصویر یا ویدیو استخراج میکنند. لایه های ادغام ابعاد فضایی ویژگیها را کاهش میدهند و لایه های کاملا متصل بر اساس ویژگیهای استخراج شده، عملیات طبقه بندی و یا رگرسیون مورد نظر را انجام میدهند. این معماری سلسله مراتبی، CNNها را قادر میسازد تا الگوهای بصری پیچیده را به طور خودکار یاد بگیرند. این قابلیت مدلهای کانولوشنی، آنها را در کارهایی مانند طبقه بندی تصویر، تشخیص اشیا و بخشبندی تصویر بسیار پرقدرت میسازد.

CNN ها در حوزههای مختلف به موفقیت های بزرگی دست یافته اند و به پیشرفت در زمینه بینایی ماشین کمک چشمگیری کردهاند. تاریخچه این شبکه ها به سالهای ۱۹۸۰ و ۱۹۹۰ برمیگردد، زمانی که با وجود فراگیر شدن رایانه ها و روشهای جدید پردازش تصویر مانند تبدیل فوریه و Transform Fourier Discrete Small و ...همچنان محدودیتهای زیادی در این حوزه وجود داشت. اما با پیدایش شبکه های کانولوشنی، این مدلها توانستند با نوآوریهای خود در زمینه تشخیص الگو و تصویر، به پردازش تصویر به صورت خودکار و با کارایی بالا بپردازند.

در این پروژه قصد داریم از شبکه عصبی CNN برای ایجاد مدلی برای طبقه بندی تصاویر تصویربرداری مغز استفاده کنیم. در این مدل وجود یا عدم وجود یا مشکوک بودن تومور مغزی در هر تصویر را برر سی کرده و در یکی از د سته بندی های مرتبط قرار می دهیم.



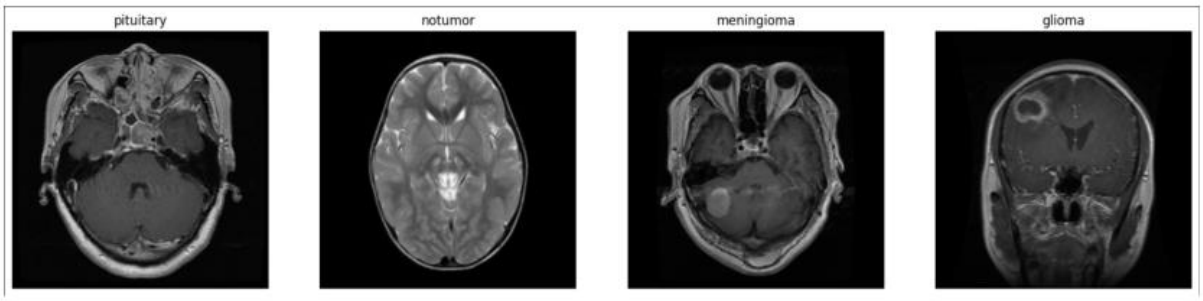
شکل ۱. معماری کلی یک CNN

تعریف مسئله

در این پروژخ، ما به پیاده سازی یک شبکه کانولوشنی برای طبقه بندی تصاویر با استفاده از فریمورک PyTorch پرداخته ایم. برای راحتی استفاده از کتابخانه ها و تسریع فرایند آموزش، از سرویسهایی مثل Colab Google و یا Kaggle استفاده کرده ایم. چون آموزش شبکه های عمیق بر روی GPU بسیار سریعتر از CPU میباشد، در این پروژه مجبور به استفاده از این سرویس ها هستیم و نوع ران تایم را در آن ها روی GPU قرار داده ایم.

.آشنایی با مجموعه داده

مجموعه داده استفاده شده در این تمرین، مجموعه تصاویر مربوط به تصاویر MRI از مغز برای تشخیص تومورهای مغزی مختلف است. این دیتاست شامل عکسهای مختلف متعلق به ۴ کلاس ، notumor meningioma، glioma و pituitary است. توجه کنید که تعداد دادههای موجود در هر کلاس با هم برابر نیستند. دیتاست مربوطه را میتوانید از (AI-CA5-Dataset.zip - Google Drive) دریافت و استفاده کنید.



پیش پردازش داده -ابتدا یک کلاس دیتاست شخصی سازی شده که از کلاس Dataset پایتورچ استفاده میکند نوشتیمو تمام تصاویر موجود در دیتاست را با استفاده از ابزارهای موجود به ابعاد ۵۱۲ در ۵۱۲ تغییر ابعاد دادیم .

گزارش کامل این بخش و دلیل استفاده یا عدم استفاده از Normalization و درصد تقسیم بندی داده test و train به طور کامل در فایل ipynb پروژه ذکر شده است.

طبقه بندی تصاویر با استفاده از شبکه CNN

در این بخش از پروژه میخواهیم شبکه خود را ساخته و سپس آن را آموزش دهیم .همانطور که توضیح داده شد، شبکه های کانولوشنی، با استفاده از لایه های کانولوشنی، لایه های ادغام و لایه های کاملا متصل، قادرند اطلاعات مربوط به الگوها و ویژگیهای مختلف در تصویر را استخراج و استفاده کنند. لایه کانولوشنی با استفاده از عملیات کانولوشن، فیلترها را بر روی تصویر اعمال کرده و نمایش جدیدی از تصویر را ایجاد میکند که شامل ویژگیهای محلی است. این لایه ها به صورت مکرر در سراسر

شبکه استفاده میشوند تا ویژگیهای سطح بالاتر را استخراج کنند .لایه های ادغام به منظور کاهش ابعاد تصویر و حذف اطلاعات بی اهمیت، استفاده میشوند. این لایه ها با استفاده از معیارهایی مانند حداکثرگیری یا میانگین گیری ، به کاهش ابعاد ویژگیهای استخراج شده میپردازند .لایه های کاملا متصل، در نهایت با استفاده از ویژگیهای استخراج شده توسط لایه های کانولوشنی و ادغام، تصمیم گیری نهایی را برای طبقه بندی تصویر انجام میدهند. این لایه ها مشابه لایه های معمولی در شبکه های عصبی عمل میکنند و خروجی نهایی را تولید میکنند که شامل احتمالهای مربوط به تعلق تصویر به هر کلاس است .دقت کنید که ورودی شبکه عصبی در این فاز، تصاویر هستند که ساختار چند بعدی دارند. همچنین ویژگیهای استخراج شده توسط لایه های کانولوشنی ما هم ساختار چند بعدی خواهند داشت ولی ورودی لایه های کاملا متصل ما به صورت یک بردار هستند. در نتیجه نیاز است که بین این دو بخش از یک الیه Flattenاستفاده کنیم.

معماری شبکه CNN

در این بخش ابتدا باید مربوط به مدل خود را با معماری مد نظر خود پیاده سازی کنید. برای این کار از ماژولها و لایه های آماده موجود در PyTorch مثل ReLU ، Linear، d2MaxPool، d2Convو Sequentialاستفاده شده است. همچنین توضیحات مرتبط به هایپرپارامترها و البته دلایل استفاده از هر کدام از ان ها به همراه تعاریف هر یک از بخش های شبکه عصبی در فایل ipynb پروژه به طور کامل ذکر شده است.

تابع هزینه

پس از تعریف مدل خود، نیاز است تا تابع هزینه مناسب برای محاسبه loss شبکه در حین آموزش و در نتیجه آموزش صحیح مدل را انتخاب کنید. دلیل استفاده از هر تابع هزینه و اقسام آن ها نیز در فایل ipynb پروژه به طور کامل ذکر شده است.

بهینه ساز

برای آموزش مدل همچنین نیاز به یک بهینه ساز داریمکه در این پروژه از بهینه ساز Adam استفاده شده است، همچنین در فایل ipynb ۷روژه به طور کامل درباره نحوه عملکرد این الگوریتم وو البته تفاوت آن با بهینه ساز SGD ذکر شده است.
همچنین برای آموزش مدل در این بخش نیاز است که یک DataLoader تعریف کرده و در فرایند آموزش از آن استفاده کنیم. همچنین درباره پارامتر size batch و تاثیر آن در فرایند آموزش توضیحات کافی در پروژه امده است .در پایان نیاز است که مدل خود را با استفاده از نتایج بخشهای قبل آموزش دهیم

پس از اتمام فرآیند آموزش نمودار loss و accuracy برحسب epoch را هم برای دادگان آموزش و هم برای دادگان ارزیابی در حین فرایند آموزش رسم کرده ایم تا تاثیر overfitting را روی داده های مشاهده کنیم. در نهایت بار دیگر مدل را با استفاده از روش های Regulariztion اجرا کرده و تفاوت ها را گزارش کرده ایم.

.تاثیر روشهای Regulartization بر روی آموزش مدل

روشهای Regularization در فرایند آموزش شبکه های عصبی، روشهایی هستند که برای جلوگیری از overfittingدر حین آموزش مورد استفاده قرار میگیرند. دو روش بسیار متداول برای انجام این کار استفاده از Dropoutو Normalization

Batchهستند. که نحوه عملکرد هر یک به طور کامل در فایل پروژه توضیح داده شده است و هر دوی این روش ها بر روی مدل اعمال شده است. در این پروژه از لایه های Dropout بین لایه های بخش Connected Fully و از لایه Normalization Batchدر میان لایه های convolutional استفاده شده است چرا که باعث ایجاد دقت بهتری و خطای کمتری خواهد شد.

ارزیابی و تحلیل نتایج

پس از آموزش مدل نوبت به ارزیابی عملکرد مدل بر روی مجموعه داده ارزیابی میرسد. که مقادیر ۴ معیار Recall ، Precision ، Accuracyو Score 1F به ازای هر کلاس و همچنین، Matrix Confusionمربوط به نتایج به دست آمده را رسم و تحلیل کرده ایم.
همان طور که در فایل پروژه قابل مشاهده است دقتی در حدود ۸۵ درصد برای داده test و در حدود ۹۳ درصد برای داده train به دست آمد که نتیجه ای قابل قبول است.

پاسخ دهی به برخی سوالات مطرح درباره عملکردهای جزئی ممکن در کد (به طور کامل در فایل پروژه توضیح داده شده)

بررسی تاثیر عملیات Normalization:

"نرمال سازی" یک مرحله پیش پردازش است که معمولاً روی مجموعه داده های تصویر قبل از فرستادن آنها به یک شبکه عصبی برای آموزش یا استنتاج اعمال می شود. این فرآیند شامل تغییر دامنه مقادیر شدت پیکسل به یک مقیاس استاندارد است. در اینجا یک طرح کلی از اینکه چگونه عادی سازی می تواند بر مجموعه داده ها و مزایا و معایب آن تأثیر بگذارد آورده شده است.

مزایای Normalization:

۱. **همگرایی (Convergence)را بهبود می بخشد**: عادی سازی به تثبیت فرآیند یادگیری کمک می کند و با اطمینان از اینکه ویژگی ها در مقیاس مشابه هستند، همگرایی را سرعت می بخشد.

۲. **تغییر کوواریت داخلی را کاهش می دهد:** با عادی سـازی ورودی ها، شـانس تغییر متغیر کمکی داخلی را کاهش می دهیم که با اجازه دادن به نرخ یادگیری بالاتر، آموزش را سرعت می بخشد.

۳. **کمک به جلوگیری از ا شباع:** شبکه های ع صبی (به ویژه آنهایی که دارای توابع فعال سازی سیگموئید ه ستند) در معرض خطر "ا شباع" هستند، جایی که مقادیر ورودی شدید می توانند گرادیان ها را در طول انتشار پس از انتشار "کشتن" کنند. عادی سازی به حفظ مقادیر ورودی در محدوده ای که نورون ها نسبت به تغییرات ورودی حساس ترین هستند، کمک می کند.

۴. **وزن های اولیه بهتر:** در طول مقدار دهی اولیه وزن، داشـتن ویژگی هایی در مقیاس مشـابه به بهینه سـاز اجازه می دهد تا از چشم انداز تلفات بهتر عبور کند و به طور بالقوه حداقل های بهتری را بیابد.

۵. **سوگیری (bias) را حذف می کند:** اگر ویژگی های خاصی در مقیاس های بزرگتر از سایرین ارائه شوند، می توانند بر فرآیند بهینه سازی تسلط داشته باشند. عادی سازی تضمین می کند که این اتفاق نمی افتد.

معایب Normalization:

۱. از دست دادن برخی اطلاعات: در مواردی که کنتراست تصویر برای درک محتوا مهم است، عادی سازی کلی ممکن است به طور غیر ضروری این جنبه را استاندارد کند.

۲. **پیچیدگی:** مراحل پردازش اضافی را در خط لوله داده اضافه می کند که گاهی اوقات ممکن است پیچیده باشد اگر تصاویر مختلف به انواع مختلف عادی سازی نیاز داشته باشند.

۳. **وابستگی پارامتر**: اگر مرحله نرمال سازی به پارامترهای خاصی (به عنوان مثال، میانگین و انحراف استاندارد مجموعه آموزشی) بستگی دارد، این پارامترها باید ذخیره شوند و در طول استنتاج نیز به درستی روی داده ها اعمال شوند و یک مرحله اضافی به آن اضافه شود. فرآیند.

۴. **مصنوعات بالقوه**: برخی از رویکردهای عادی سازی ساده یا تهاجمی ممکن است مصنوعاتی را در تصاویر ایجاد کنند که قبل از پردازش وجود نداشتند، که می تواند بر آموزش مدل تأثیر بگذارد.

۵. **نیاز به دانش دامنه**: ممکن است مواردی وجود داشته باشد که برای تصمیم گیری اینکه چه نوع عادی سازی مناسب است به دانش خاص دامنه نیاز باشد (مثلاً عادی سازی محلی در مقابل جهانی)، در غیر این صورت در صورت انتخاب نادرست می تواند بر مدل تأثیر منفی بگذارد.

عادی سازی تعداد تصاویر موجود در مجموعه داده را تغییر نمی دهد. فقط مقادیر پیکسل های درون آن تصاویر را تغییر می دهد. تأثیر روی مجموعه داده بر شکل و کیفیت داده ها برای کارایی محاسباتی و یادگیری است، نه بر اندازه مجموعه داده.

با این حال، اعمال پارامترهای عادی سازی یکسان (به عنوان مثال، همان میانگین و انحراف استاندارد که برای تصاویر آموزشی استفاده می شود) برای اعتبارسنجی و مجموعه های آزمایش بسیار مهم است تا شرایط را در کل خط لوله داده ثابت نگه دارید و از نشت داده ها جلوگیری کنید، بنابراین اطمینان حاصل شود که آموزش دیده مدل با داده های جدید به طور دقیق عمل می کند.

پس از آزمایش هر دو روش با و بدون نرمال سازی، مشخص شد که بدون نرمال سازی، دقت بیشتری حاصل می شود، زیرا داده های از دست رفته پس از کانولوشن به طور طبیعی وجود دارد. با عادی سازی، مقدار داده های از دست رفته بیشتر و بیشتر می شود.

در مورد پارامتر اندازه دسته (batch size) و تاثیر آن در فرآیند آموزش توضیح دهید

اندازه دسته ای (batch size) یک فراپارامتر است که تعداد نمونه هایی را که باید قبل از به روز رسانی پارامترهای مدل داخلی کار کنند، تعیین می کند. هنگام آموزش یک شبکه عصبی، به روز رسانی وزن مدل با هر نمونه (یادگیری آنلاین) یا پس از بررسی کل مجموعه داده (به دلیل بار محاسباتی و محدودیت های حافظه) امکان پذیر یا کارآمد نیست. بنابراین، آموزش اغلب با استفاده از مینی بچ (mini-batch) انجام می شود.

اثرات اندازه دسته (batch) بر فرآیند آموزش:

۱. استفاده از حافظه:

- دسته های کوچکتر از حافظه کمتری استفاده می کنند که به شما امکان می دهد مدل را با استفاده از منابع محدود آموزش دهید.

- دسته های بزرگتر حافظه بیشتری مصرف می کنند، اما می توانید از روال های ضرب ماتریس بسیار بهینه شده استفاده کنید.

۲. عملکرد مدل و تعمیم:

- دسته های کوچکتر می توانند نویز را به فرآیند آموزش وارد کنند که می تواند منجر به تعمیم قوی تر شود. با این حال، آنها همچنین باعث می شوند که آموزش پایدارتر نباشد.

- دسته های بزرگتر گرادیان خطای پایدارتری برای بهینه ساز فراهم می کنند که شاید به همگرایی پایدارتر منجر شود، اما ممکن است به حداقل سازهای تیز همگرا شوند و در نتیجه منجر به تعمیم ضعیف تر شوند.

۳. کارایی محاسباتی:

- دسته های کوچکتر می تواند منجر به کاوش کامل تر از چشم انداز خطا شود، در حالی که احتمالاً زمان بیشتری برای همگرایی به حداقل می رسد.

- دسته های بزرگتر باعث می شود که هر تکرار بیشتر طول بکشد، اما ممکن است مدل برای همگرایی به تکرارهای کمتری نیاز داشته باشد.

۴. سرعت همگرایی:

- دسته های کوچکتر گاهی اوقات می توانند به دلیل نویز در تخمین گرادیان خود از حداقل های محلی فرار کنند که به طور بالقوه منجر به راه حل های کلی بهتر می شود.

- دسته های بزرگتر دارای یک اثر صاف کننده هستند که گاهی اوقات باعث می شود بهینه ساز در حداقل محلی گیر کند.

۵. استفاده از موازی سازی:

- دسته‌های بزرگ‌تر معمولاً بیشتر در معرض موازی‌سازی هستند و امکان استفاده کارآمدتر از CPUها و GPUهای چند هسته‌ای را فراهم می‌کنند که می‌تواند به طور کلی به زمان‌های آموزشی سریع‌تر منجر شود.

انتخاب اندازه دسته مناسب:

- اغلب نیاز به آزمایش دارد زیرا اندازه دسته بهینه می تواند بسته به مجموعه داده خاص و معماری شبکه عصبی متفاوت باشد.

- به عنوان یک قاعده کلی، قدرت های ۲ اغلب انتخاب می شوند (۳۲، ۶۴، ۱۲۸، ۲۵۶، ...) به دلیل نحوه ساختار و دسترسی به حافظه در معماری های محاسباتی مدرن، که امکان بازیابی کارآمدتر داده ها را فراهم می کند.

ملاحظات دیگر:

- اندازه دسته در مقابل نرخ یادگیری: اندازه های دسته ای بزرگ ممکن است به نرخ یادگیری متفاوتی نسبت به اندازه های دسته ای کوچک نیاز داشته باشند. در برخی موارد، اندازه‌های دسته‌ای بزرگ‌تر نشان داده شده است که از مقیاس‌بندی نرخ یادگیری سود می‌برند (مقیاس‌سازی خطی نرخ یادگیری با اندازه دسته‌ای).

- پایداری Training: اندازه های بسیار بزرگ دسته ای ممکن است باعث بی ثباتی Training شود، به خصوص با مقیاس ناکافی نرخ یادگیری.

- در عمل، بسیاری از محققین و پزشکان، اندازه دسته را با حافظه موجود سخت افزار مورد استفاده برای آموزش، همراه با آزمایشات تجربی برای یافتن اندازه ای که سازش خوبی بین سرعت تمرین و عملکرد مدل ایجاد می کند، متعادل می کنند.

نحوه کار و همچنین دلیل استفاده از موارد زیررا توضیح دهید. موارد مورد نیاز در این بخش شامل اندازه هسته و stride (گام) برای لایه های مختلف، padding استفاده شده در لایه های مختلف، تعداد لایه های کانولوشن و عمق شبکه کانولوشن است.

لایه های کانولوشنال:

لایه های کانولوشن بلوک های اصلی یک CNN هستند. آنها مجموعه ای از فیلترهای قابل یادگیری را روی تصویر ورودی برای ایجاد نقشه های ویژگی اعمال می کنند. این فیلترها سلسله مراتب فضایی ویژگی ها را شناسایی می کنند - از لبه ها در لایه های اولیه تا الگوهای پیچیده در لایه های عمیق تر.

اندازه هسته:

اندازه هسته (یا فیلتر) عرض و ارتفاع پنجره فیلتر را تعیین می کند که روی تصویر ورودی می لغزد تا یک پیکسل را در نقشه ویژگی ایجاد کند. به عنوان مثال، یک فیلتر x۳۳ به ناحیه ۳ پیکسل در ۳ پیکسل از تصویر نگاه می کند.

- چرا از اندازه های مختلف هسته استفاده کنیم؟

- هسته های کوچکتر (مانند x۳۳ یا x۵۵) برای ثبت ویژگی های دقیق تر خوب هستند و از نظر محاسباتی کارآمدتر هستند.

- هسته های بزرگتر (مانند x۷۷ یا بیشتر) ویژگی های وسیع تری از تصویر را ثبت می کنند اما ابعاد فضایی را به میزان قابل توجهی کاهش می دهند و ممکن است جزئیات دقیق تری را از دست بدهند.

گام های بلند برداشتن:

Stride تعداد پیکسل‌هایی را که پنجره فیلتر بر روی تصویر ورودی حرکت می‌کند، دیکته می‌کند. گام ۱ فیلتر را هر بار یک پیکسل حرکت می دهد و منجر به نگاشت ویژگی متراکم می شود. یک گام بزرگتر، ابعاد فضایی نقشه ویژگی خروجی را کاهش می دهد، که منجر به استفاده کمتر از محاسبات و حافظه می شود، اما احتمالاً برخی از الگوهای ریز دانه را از دست می دهد.

لایه گذاری:

Padding شامل افزودن لایه‌هایی از صفر در اطراف مرز تصویر ورودی است تا عملیات کانولوشن روی عناصر مرزی تصویر ورودی اعمال شود.

- چرا از padding استفاده کنیم؟

- برای کنترل اندازه فضایی حجم های خروجی، اغلب برای حفظ ابعاد ورودی و خروجی یکسان

- برای فعال کردن شبکه برای یادگیری ویژگی ها از گوشه ها و لبه های تصاویر.

تعداد لایه های کانولوشنال:

این به عمق شبکه، تعداد لایه های کانولوشنال که روی هم چیده شده اند، اشاره دارد. افزایش عمق می تواند توانایی شبکه را برای نمایش ویژگی های پیچیده افزایش دهد.

- چرا تعداد لایه ها را تنظیم کنید؟

- لایه های بیشتر می تواند به شبکه کمک کند تا نمایش سلسله مراتبی داده ها را بیاموزد.

- لایه های بیش از حد ممکن است منجر به نصب بیش از حد شود و آموزش شبکه را سخت تر کند، به خصوص بدون داده های کافی.

- هرچه عمیق تر می شوید، اندازه فضایی کاهش می یابد در حالی که عمق (تعداد فیلترها) افزایش می یابد و از پیکسل های خام به مفاهیم انتزاعی حرکت می کند.

عمق شبکه کانولوشن:

اصطلاح "عمق" می تواند به دو چیز اشاره داشته باشد: تعداد لایه ها یا تعداد کانال ها (نقشه های ویژگی) در هر لایه.

- چرا عمق را افزایش دهیم؟

- شبکه های عمیق تر به طور بالقوه می توانند ویژگی های پیچیده تری را یاد بگیرند و به عملکرد بهتری دست یابند.

- با این حال، به دلیل مسائلی مانند ناپدید شدن شیب ها، آموزش آنها می تواند چالش برانگیز باشد، اگرچه تکنیک هایی مانند پرش از اتصالات (به عنوان مثال، ResNet) و عادی سازی دسته ای به غلبه بر این چالش ها کمک می کنند.

- کانال های عمیق تر (فیلترهای بیشتر) در یک لایه به شبکه اجازه می دهد تا طیف گسترده تری از ویژگی ها را از همان سطح پیچیدگی در داده ها یاد بگیرد.

ملاحظات: انتخاب اندازه هسته، گام برداشتن، و padding نیز با اندازه تصویر و تعداد لایه های کانولوشن برای تعیین معماری کلی شبکه در تعامل است. اندازه و ساختار مناسب شبکه به همان اندازه که یک علم است هنر است و اغلب شامل تست و تنظیم

تجربی است. هنگام طراحی یک معماری CNN، شما اغلب با استانداردهای صنعتی یا معماری های شبکه اثبات شده (مانند VGG، ResNet، Inception، و غیره) شروع می کنید و سپس بر اساس مشکل خاص و اندازه مجموعه داده خود سفارشی می کنید و تطبیق می دهید. هدف ایجاد تعادل بین عملکرد (از لحاظ کیفیت پیش‌بینی و کارایی محاسباتی) با اجتناب از برازش بیش از حد، تحت محدودیت‌های مربوط به آنچه برای داده‌های شما و مشکل موجود است، است. تنظیم فراپارامتر و جستجوی معماری شبکه زمینه های گسترده ای هستند که به دنبال خودکارسازی این بخش از توسعه مدل هستند.

پارامترهای nn.Conv2d:

۱. **"num_channels"**: تعداد کانال های ورودی برای اولین لایه کانولوشن. برای تصاویر RGB، این ۳ است.

۲. «**out_channels**»: تعداد فیلترها یا هسته‌هایی که لایه کانولوشنال از ورودی یاد می‌گیرد. در اینجا، روی ۱۶ تنظیم شده است، یعنی ۱۶ نقشه ویژگی بعد از این لایه وجود خواهد داشت.

۳. `kernel_size`: اندازه فیلتر کانولوشن. «اندازه_کرنل» ۱۰ به معنای فیلتر ۱۰×۱۰ است.

۴. «**padding**»: لایه صفر اضافه شده به مرزهای ماتریس ورودی. 1 padding یک لایه ۰ به اطراف تصویر اضافه می کند که به هسته امکان می دهد لبه های تصویر ورودی را پردازش کند.

nn.ReLU:

این یک تابع فعال سازی است که غیرخطی بودن را به مدل وارد می کند و به آن امکان می دهد الگوهای پیچیده تری را بیاموزد. تابع ReLU به صورت "f(x) = max(0, x)" تعریف می شود.

nn.AvgPool2d:

۱. `kernel_size`: اندازه پنجره ای که مقدار متوسط روی آن گرفته می شود. در اینجا، x2۲ است، به این معنی که ارتفاع و عرض نقشه های ویژگی را ضریب ۲ کاهش می دهد.

۲. "گام": میزان حرکت پنجره ادغام را برای هر مرحله ادغام کنترل می کند. در این مورد نیز ۲ است، به این معنی که بین مناطق ادغام شده همپوشانی وجود ندارد.

لایه های کاملا متصل – nn.Linear:

۱. «self._to_linear»: این یک پارامتر استاندارد PyTorch نیست، بلکه یک مقدار تعریف شده توسط کاربر است که بر اساس شکل خروجی «conv_layer» محاسبه شده است. به روز رسانی این محاسبه بسیار مهم است زیرا عملیات پیچیدگی و ادغام اندازه نقشه ویژگی خروجی را تغییر می دهد. برای کانولوشن با بالشتک ۱ و اندازه هسته ۱۰، در واقع باید ابعاد خروجی را دوباره محاسبه کنید.

۲. "واحد_های_مخفی": تعداد نورون ها در لایه پنهان قسمت کاملاً متصل. روی ۲۵۶ تنظیم شده است و یک هایپرپارامتر است که می توان آن را تنظیم کرد.

۳. "output_shape": اندازه لایه خروجی که با تعداد کلاس‌های یک کار طبقه‌بندی مطابقت دارد. در اینجا، روی ۴ تنظیم شده است.

پاس رو به جلو (forward pass):

روش "به جلو" نحوه حرکت داده های شما در شبکه را مشخص می کند. ورودی «x» از لایه‌های کانولوشنال عبور می‌کند، سپس برای مطابقت با ابعاد مورد انتظار لایه‌های کاملاً متصل، مسطح می‌شود و در نهایت، از این لایه‌های خطی عبور داده می‌شود تا خروجی تولید شود.

اثرات تغییر پارامترها و فراپارامترها:

۱. **افزایش «num_channels»**: این معمولاً بر اساس داده‌های ورودی است (به عنوان مثال، تصاویر RGB دارای ۳ کانال، مقیاس خاکستری دارای ۱ کانال). تغییر خودسرانه آن منطقی نخواهد بود.

۲. **"out_channels" بیشتر**: فیلترهای بیشتر ویژگی های اضافی را جذب می کنند اما هزینه محاسباتی و پیچیدگی مدل را افزایش می دهند.

۳. **«اندازه_هسته» بزرگتر**: هسته های بزرگتر زمینه بیشتری را ثبت می کنند اما اندازه نقشه ویژگی را به شدت کاهش می دهند.

۴. افزایش «padding»: مفید برای حفظ اندازه نقشه ویژگی، می تواند منجر به استخراج بهتر ویژگی لبه شود. بالشتک بیش از حد می تواند شدت ویژگی را رقیق کند.

۵. **"hidden_units"** بیشتر: تعداد بیشتر به شبکه اجازه می دهد تا الگوهای پیچیده تری را بیاموزد، اما ممکن است به بیش از حد برازش و افزایش هزینه های محاسباتی منجر شود.

۶. **تغییر 'output_shape'**: این بر اساس تعداد کلاس های خروجی شما است. تغییر آن در غیر این صورت با الزامات مشکل مطابقت ندارد.

علاوه بر این:

- تعداد لایه: افزودن لایه های بیشتر می تواند به شبکه کمک کند تا الگوهای عمیق تر و انتزاعی تر را بیاموزد. با این حال، می تواند منجر به بیش از حد برازش و افزایش تقاضای محاسباتی شود. این یک تعادل ظریف است و باید با عملکرد اعتبار سنجی متقابل مطلع شود.

- اندازه دسته: اندازه دسته بزرگتر منجر به تخمین گرادیان پایدارتر در طول آموزش می شود اما می تواند باعث مشکلات حافظه شود. اندازه‌های دسته‌ای کوچکتر به‌روزرسانی‌های بیشتری را در هر دوره ارائه می‌کنند، اما می‌توانند نویز داشته باشند.

- نرخ یادگیری: اندازه مراحل به روز رسانی را در حین بهینه سازی کنترل می کند. نرخ یادگیری بالا ممکن است به سرعت همگرا شـــود، اما می تواند از حداقل ها فراتر رود، در حالی که نرخ یادگیری پایین ممکن اســـت در حداقل های محلی گیر کند یا زمان زیادی طول بکشد تا همگرا شوند.

به یاد داشته باشید که تنظیم این پارامترها می تواند به طور قابل توجهی بر میزان خوب یادگیری مدل شما از داده های آموزشی، ســـرعت آموزش و تعمیم آن به داده های دیده نشـــده تأثیر بگذارد. برای تعیین مقادیر بهینه برای کار و مجموعه داده خاص خود، اغلب آزمایش و استفاده از تکنیک هایی مانند اعتبار سنجی متقاطع ضروری است.

همانطور که مشاهده می شود، وقتی از روش های regularization استفاده می کنیم، مدل خیلی دیرتر overfit می شود. این روش ها به کاهش ضرایب بسیار بزرگ در مدل کمک می کنند و احتمال پیچیدگی بسیار زیاد آن کمتر است. پس از انجام این روش نتایج زیر را به دست خواهید آورد که از نتایج بدون regularization کمی بهتر است

توضیحات مفصل تر از هر الگوریتم، نحوه پیاده سازی و جزئیات پیاده سازی و توضیحات هر پارامتر و هر انتخاب به طول کامل در فایل پروژه که به پیوست ارسال شده است موجود است.

لینک های مفید:

Learn the Basics — PyTorch Tutorials 2.2.0+cu121 documentation

Google Colab Tutorial (tutorialspoint.com)

Loss Functions in PyTorch Models - MachineLearningMastery.com

Gentle Introduction to the Adam Optimization Algorithm for Deep Learning - MachineLearningMastery.com

How to Avoid Overfitting in Deep Learning Neural Networks - MachineLearningMastery.com